**FINAL PROJECT REPORT**

**Book a doctor using MERN**

***Submitted by***

**Team Members**

**Varanasi Hemasai Reddy – 21MIM10056**

**Abhishek Gangwar – 22BCE10266**

**Mohit Aggarwal – 22BCE11488**

**Pankh Bansal –22BHI10109**

**Index**

|  |  |
| --- | --- |
| **Content** | **Page no** |
| **1. INTRODUCTION**  **1.1 Project Overview**  **1.2 Purpose** | **3** |
| **2. IDEATION PHASE**  **2.1 Problem Statement**  **2.2 Empathy Map Canvas**  **2.3 Brainstorming** | **4-6** |
| **3. REQUIREMENT ANALYSIS**  **3.1 Customer Journey map**  **3.2 Solution Requirement**  **3.3 Data Flow Diagram**  **3.4 Technology Stack** | **7-11** |
| **4. PROJECT DESIGN**  **4.1 Problem Solution Fit**  **4.2 Proposed Solution**  **4.3 Solution Architecture** | **12-13** |
| **5. PROJECT PLANNING & SCHEDULING**  **5.1 Project Planning** | **14** |
| **6. FUNCTIONAL AND PERFORMANCE TESTING**  **6.1 Performance Testing** | **15-17** |
| **7. RESULTS** | **18-22** |
| **7.1 Output Screenshots** |
| **8. ADVANTAGES & DISADVANTAGES** | **23** |
| **9. CONCLUSION** | **24** |
| **10. FUTURE SCOPE** | **25** |

## 

## **1. INTRODUCTION**

### **1.1 Project Overview**

This project focuses on developing a Doctor Appointment Booking System using the MERN stack — MongoDB, Express.js, React.js, and Node.js. The objective is to create a full-fledged web application that facilitates seamless communication between patients and doctors, enabling real-time appointment scheduling, user authentication, and role-based access.

The system allows:

* Patients to register, log in, search for doctors based on specialization, and book available slots.
* Doctors to manage their profiles, view appointment requests, and respond accordingly.
* Admins to verify doctor accounts, oversee platform activity, and maintain data integrity.

### **1.2 Purpose**

This project's primary purpose is to solve a real-world problem and serve as an end-to-end learning experience in full-stack development.

**Learning Objectives:**

* Understand and implement the MERN architecture.
* Design modular backend APIs using Express.js and Node.js.
* Integrate MongoDB as a NoSQL database for scalable data storage.
* Develop responsive and interactive frontend components using React.js.
* Use JWT for secure authentication and implement protected routes.

**Problem-Solving Objectives:**

* Streamline the healthcare appointment process using digital technology.
* Reduce administrative overhead for clinics and hospitals.
* Provide a user-friendly platform for patients to manage health consultations.
* Offer a secure and verified channel for doctor-patient interactions.

## 

## **2. IDEATION PHASE**

### **2.1 Problem Statement**

The conventional method of booking medical appointments is often time-consuming and inefficient and lacks transparency. Patients typically have to visit clinics physically or make phone calls to book appointments, leading to:

* Long waiting periods and scheduling conflicts.
* Inconvenience due to lack of information on doctor availability.
* Missed or forgotten appointments due to poor notification systems.

From the doctor’s perspective, there is a lack of tools to manage appointments effectively, resulting in:

* Overbooked or underutilized schedules.
* Difficulty maintaining patient interaction records.
* Limited accessibility for remote or follow-up consultations.

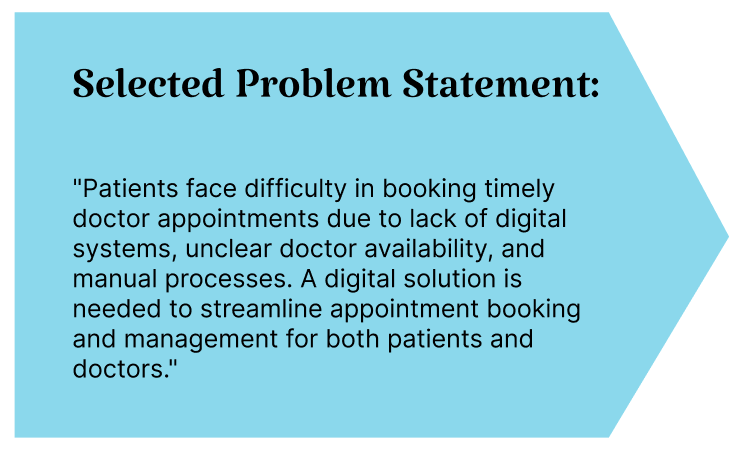
There is a growing need for a digital platform that connects patients and doctors in a streamlined and efficient manner, reducing manual effort and improving the overall healthcare experience.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Problem Statement (PS)** | **I am (Customer)** | **I’m trying to** | **But** | **Because** | **Which makes me feel** |
| PS-1 | A patient looking to consult a doctor | Book a Medical Appointment online | Most clinics  don’t have  digital  platforms or  real-time  availability | The current  process is  Manual,  Time-  Consuming and unclear | Frustrated, anxious, and unsure about when or how I’ll be treated |
| PS-2 | |  |  | | --- | --- | | A doctor trying to manage patient consultations | | |  | | Maintain and  organize  Appointments  efficiently | Patients  often miss  appointments | There is no integrated scheduling or approval system | Disorganized and stressed, unable to manage time properly |

### **2.2 Empathy Map Canvas**

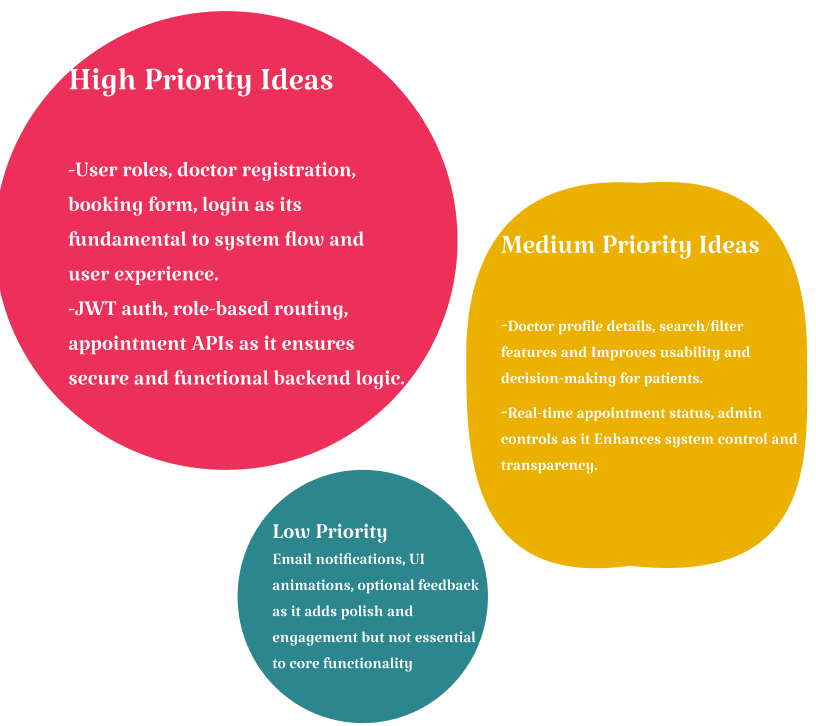
### **2.3 Brainstorming**

**Step 1: Team Gathering, Collaboration, and Selecting the Problem Statement**



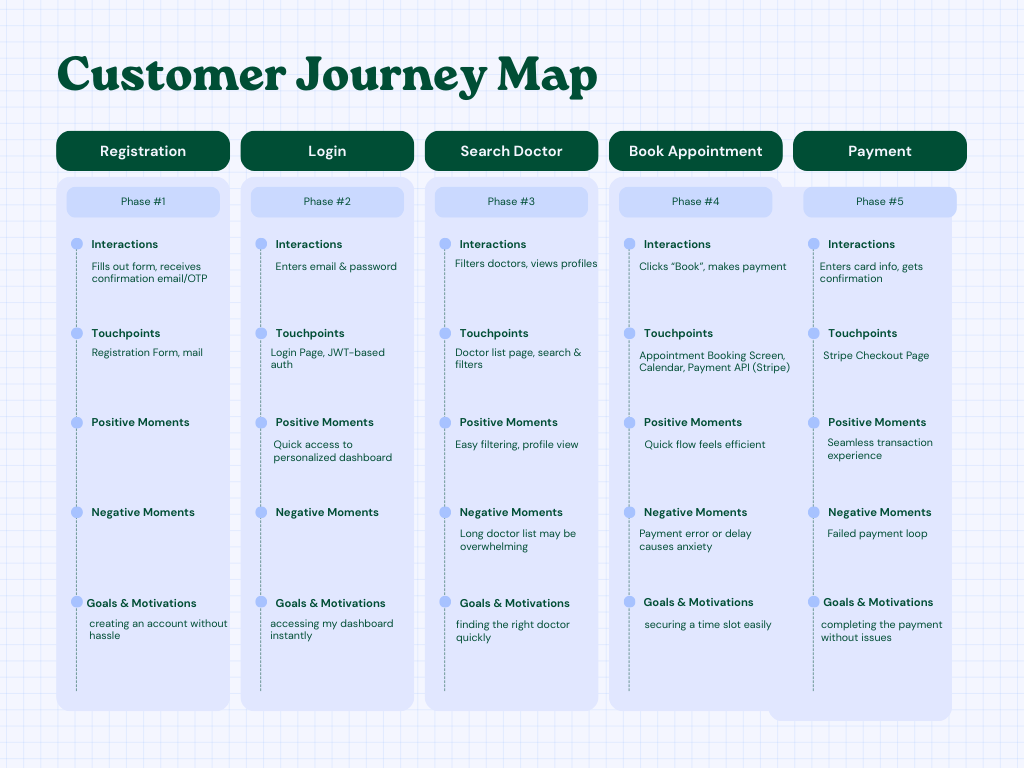
|  |  |
| --- | --- |
| **IDEA** | **GROUP** |
| Allow search/filter of doctors by specialization | User Convenience |
| Create user roles: Patient, Doctor, Admin | Core Functionality |
| Build doctor profiles with rating, availability, and bio | Transparency |
| Implement appointment booking with slot selection | Booking System |
| Use JWT for authentication and role-based access | Security |
| Admin approval for doctor registration | System Integrity |
| Real-time status updates for appointments | Communication |

**Step-2: Idea Prioritization**



**3. Requirement Analysis**

**3.1 Customer Journey Map**

****

**3.2 Solution Requirements**

**Functional Requirements:**

The following are the functional requirements of the proposed solution.

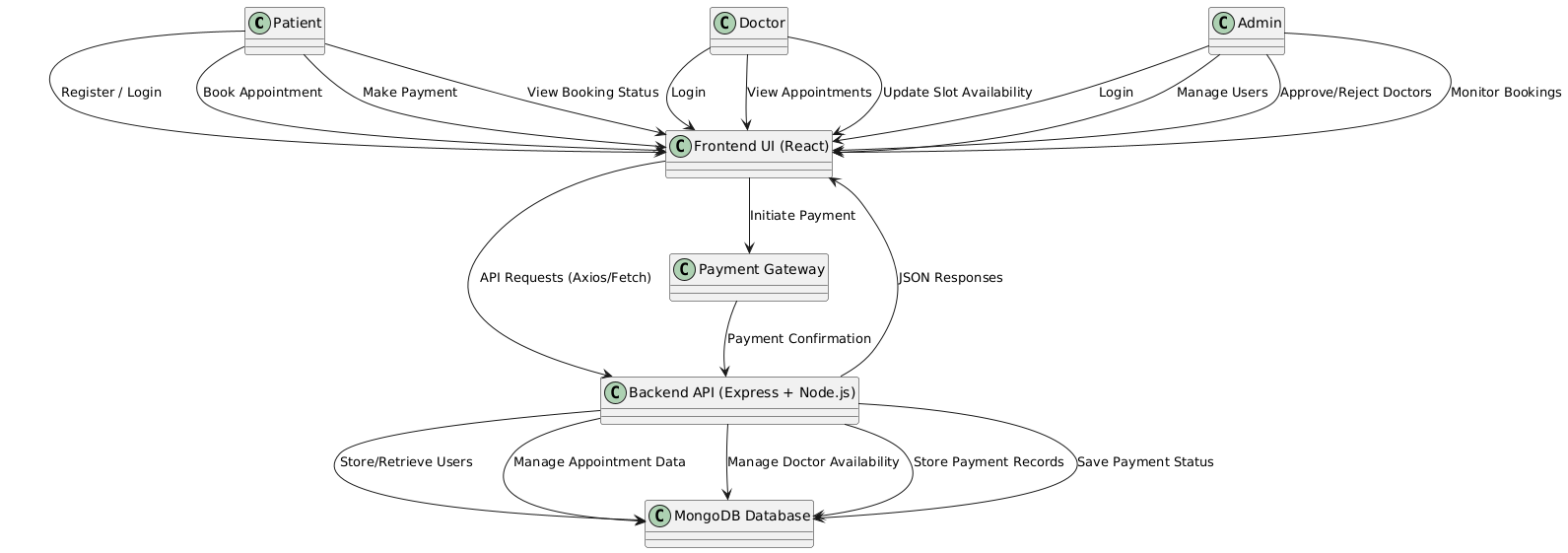
|  |  |  |
| --- | --- | --- |
| **FR No.** | **Functional Requirement (Epic)** | **Sub Requirement (Story / Sub-Task)** |
| FR-1 | User Registration & Login | Registration through form and mail |
| FR-2 | User Confirmation | NA |
| FR-3 | Role-Based Dashboard | Patient Dashboard: Search Doctors, Book Appointments  Doctor Dashboard: Manage Availability, View Appointments  Admin Dashboard: Approve Doctors, Manage Users |
| FR-4 | Appointment Booking | Select Doctor and Slot  Book Appointment  View Booking Status |
| FR-5 | Payment Integration | Initiate Payment via Gateway (Stripe)  Confirm Payment  Link Payment Status with Appointment |
| FR-6 | User Management by Admin | Add/Remove/Deactivate Users  Verify Doctor Credentials  Monitor Platform Activity |

**Non-functional Requirements:**

The following are the non-functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Non-Functional Requirement** | **Description** |
| NFR-1 | **Usability** | Clean, intuitive, and responsive user interface for all roles |
| NFR-2 | **Security** | Secure login with JWT, encrypted passwords, and HTTPS |
| NFR-3 | **Reliability** | Account security and reliable booking flow |
| NFR-4 | **Performance** | Fast API response, optimized database queries, and efficient backend |
| NFR-5 | **Availability** | Hosted on reliable platforms with minimal downtime |
| NFR-6 | **Scalability** | Easily extendable to support more users |

**3.3 Data Flow Diagram**

****

**User Stories:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| User Type | Functional Requirement (Epic) | User Story Number | User Story / Task | Acceptance Criteria | Priority | Release |
| Patient (Web user) | Registration | USN-1 | As a user, I can register for the application by entering my email, password, and confirming my password. | I can access my account/dashboard | High | Sprint-1 |
| Patient (Web user) | Registration Confirmation | USN-2 | As a user, I will receive confirmation email/OTP after registration. | I can confirm registration and activate my account | High | Sprint-1 |
| Patient (Web user) | Social Login | USN-3 | As a user, I can register via Gmail or LinkedIn. | I can register and access the dashboard using social login | Medium | Sprint-2 |
| Patient (Web user) | Login | USN-4 | As a user, I can log into the system using my email and password. | I am able to access the dashboard after login | High | Sprint-1 |
| Patient (Web user) | Appointment Booking | USN-5 | As a patient, I can search for doctors and book appointments from available slots. | I can view and confirm my booked appointments | High | Sprint-2 |
| Patient (Web user) | Payments | USN-6 | As a patient, I can pay through a payment gateway while booking an appointment. | I receive payment confirmation and appointment is confirmed | High | Sprint-2 |
| Doctor (Web user) | Login | USN-7 | As a doctor, I can log into the system securely. | I am redirected to my doctor dashboard | High | Sprint-1 |
| Doctor (Web user) | Slot Management | USN-8 | As a doctor, I can update my availability slots. | My available times are visible to patients | High | Sprint-2 |
| Doctor (Web user) | Appointment Management | USN-9 | As a doctor, I can view and manage booked appointments. | I can see a list of confirmed patient bookings | High | Sprint-2 |
| Admin (Web user) | Login | USN-10 | As an admin, I can securely log into the system. | I can access the admin dashboard | High | Sprint-1 |
| Admin (Web user) | Doctor Approval | USN-11 | As an admin, I can approve or reject doctor registration requests. | Only approved doctors are added to the system | High | Sprint-2 |
| Admin (Web user) | User Management | USN-12 | As an admin, I can manage all users (patients and doctors). | I can add, remove, or deactivate users | High | Sprint-2 |
| Admin (Web user) | Booking Monitoring | USN-13 | As an admin, I can view all appointment bookings. | I can monitor the booking activity and statistics | Medium | Sprint-2 |

**3.4 Technology Stack**

|  |  |  |  |
| --- | --- | --- | --- |
| S.No | Component | Description | Technology |
| 1 | User Interface | How users interact with the application | HTML, CSS, tailwind@3, JavaScript, React.js |
| 2 | Application Logic-1 | Handles user authentication and role-based access | Node.js, Express.js |
| 3 | Application Logic-2 | Manages appointment booking, slot availability | Node.js, Express.js |
| 4 | Application Logic-3 | Integrates payment processing and confirmation | Node.js, Express.js |
| 5 | Database | Stores user data, appointment records, and payment info | MongoDB, NoSQL, Mongoose |
| 6 | External API-1 | Payment Gateway integration for appointment payments | Stripe API |

**4. PROJECT DESIGN**

**4.1 Problem Solution Fit**

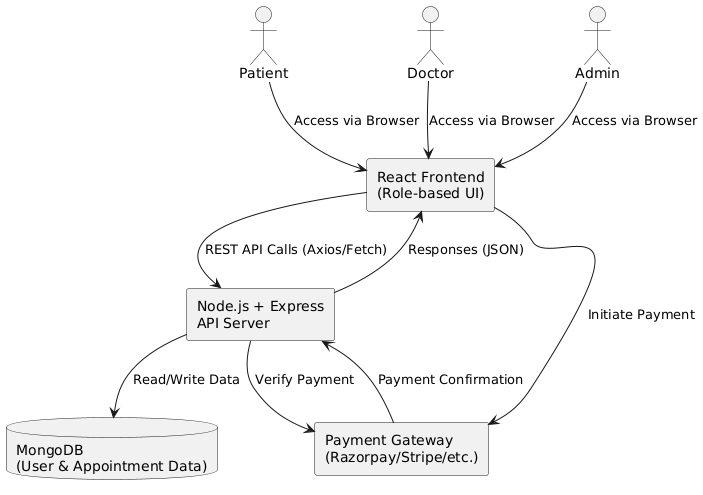
**Problem:**



**4.2 Proposed Solution**

|  |  |  |
| --- | --- | --- |
| S.No. | Parameter | Description |
| 1 | Problem Statement (Problem to be solved) | Lack of a centralized, efficient, and user-friendly platform for patients to book doctor appointments and for doctors to manage their availability and schedule. |
| 2 | Idea / Solution description | A MERN stack-based doctor appointment booking system with role-based dashboards for patients, doctors, and admins. Features include appointment booking, availability management, secure login, and payment integration. |
| 3 | Novelty / Uniqueness | Custom-built platform tailored to specific hospital/clinic needs. Role-based access and flexibility with Gmail login, integrated payments, and admin control. |
| 4 | Social Impact / Customer Satisfaction | Reduces waiting times, increases accessibility to healthcare, especially in rural areas. Enhances user convenience and healthcare system efficiency. |
| 5 | Business Model (Revenue Model) | Freemium model: Basic features for free. Premium subscription for clinics/doctors. Possible ad integration or commission per appointment booked. |
| 6 | Scalability of the Solution | Highly scalable using cloud-based infrastructure. Can be expanded to include teleconsultation, pharmacy integration, and multi-language support. |

**4.3 Solution Architecture**

****

**5. PROJECT PLANNING & SCHEDULING**

**5.1 Project Planning**

**Product Backlog, Sprint Schedule, and Estimation**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Sprint | Functional Requirement (Epic) | User Story Number | User Story / Task | Story Points | Priority | Team Members |
| Sprint-1 | Registration | USN-1 | As a user, I can register by entering my email and confirming my password. | 2 | High | Abhishek Gangwar |
| Sprint-1 | Login | USN-4 | As a user, I can log in using my email and password. | 1 | High | Abhishek Gangwar |
| Sprint-2 | Appointment Booking | USN-5 | As a patient, I can search doctors and book appointments. | 3 | High | Varanasi Hemasai Reddy |
| Sprint-2 | Payments | USN-6 | As a patient, I can make payment via gateway to confirm appointment. | 3 | High | Varanasi Hemasai Reddy |
| Sprint-3 | Doctor Dashboard | USN-7 | As a doctor, I can manage my availability slots. | 2 | High | Mohith Aggarwal |
| Sprint-3 | Doctor Appointment View | USN-8 | As a doctor, I can view all booked appointments. | 2 | High | Mohith Aggarwal |
| Sprint-4 | Admin Login | USN-9 | As an admin, I can log in and access the admin dashboard. | 1 | High | Pankh Bansal |
| Sprint-4 | Admin Doctor Approval | USN-10 | As an admin, I can approve or reject doctor registrations. | 2 | High | Pankh Bansal |

**Project Tracker, Velocity & Burndown Chart**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Sprint | Total Story Points | Duration | Sprint Start Date | Sprint End Date (Planned) | Story Points Completed | Sprint Release Date (Actual) |
| Sprint-1 | 4 | 2 Days | 01 Apr 2025 | 02 Apr 2025 | 4 | 02 Apr 2025 |
| Sprint-2 | 8 | 2 Days | 03 Apr 2025 | 04 Apr 2025 | 8 | 04 Apr 2025 |
| Sprint-3 | 4 | 2 Days | 05 Apr 2025 | 06 Apr 2025 | 4 | 06 Apr 2025 |
| Sprint-4 | 6 | 2 Days | 07 Apr 2025 | 08 Apr 2025 | 6 | 08 Apr 2025 |

**6. FUNCTIONAL AND PERFORMANCE TESTING**

**6.1 Performance Testing**

**Project Overview:**

**Project Name:** BOOK A DOCTOR USING MERN

**Project Description:**

Book a doctor is a web application built using the MERN stack (MongoDB, Express.js, React.js, Node.js). It allows patients to register, log in, search for doctors, and book appointments online. Doctors can manage their availability and view scheduled appointments. Admins oversee user management and approve doctor profiles. The app includes secure login and Stripe payment integration for confirmed bookings.

**Project Version:** "1.0.0"

**Testing Period:** 09-04-2025 to 10-04-2025

**Testing Scope:**

**List of Features and Functionalities to be Tested:**

* User Registration via Email and Gmail
* User Login and Role-based Dashboard Redirection
* Doctor Search and Appointment Booking by Patients
* Payment Integration and Confirmation via Stripe
* Doctor Availability Management
* Viewing Appointments (Doctor and Patient Dashboards)
* Admin Functions: Approving Doctors, Managing Users

**List of User Stories or Requirements to be Tested:**

* Email Registration
* Login Functionality
* Appointment Booking
* Payment Confirmation
* Doctor Availability Updates
* Doctor Appointment Views
* Admin Login
* Admin Approves Doctors
* Admin User Management

**Testing Environment:**

URL/Location: <https://localhost:5173/>

Credentials (if required):

**Doctor:**

User: [arjunmehta@gmail.com](mailto:arjunmehta@gmail.com)

Password: arjunmehta

**Patient:**

User: [user1@gmail.com](mailto:user1@gmail.com)

Password: user1

**Test Cases:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Scenario** | **Test Steps** | **Expected Result** | **Actual Result** | **Pass/Fail** |
| TC-001 | User Registration with Email | 1. Open Registration Page  2. Enter Name, Email, Password  3. Click Register | User is registered and redirected to confirmation message | User successfully registered | Pass |
| TC-002 | Login with Valid Credentials | 1. Open login page  2. Enter email and password  3. Click Login | User is redirected to the correct dashboard based on role | Correct dashboard shown | Pass |
| TC-003 | Book Appointment as Patient | 1. Login as Patient  2. Search for doctor  3. Select slot and click Book  4. Make Payment | Appointment is booked and payment confirmed | Booking confirmed | Pass |
| TC-004 | View Appointments as Doctor | 1. Login as Doctor  2. Navigate to "My Appointments" | List of upcoming appointments is displayed | Appointments listed | Pass |
| TC-005 | Admin Approves Doctor Registration | 1. Login as Admin  2. Go to pending doctor approvals  3. Approve a pending doctor | Doctor account status changes to active | Approval processed | Pass |

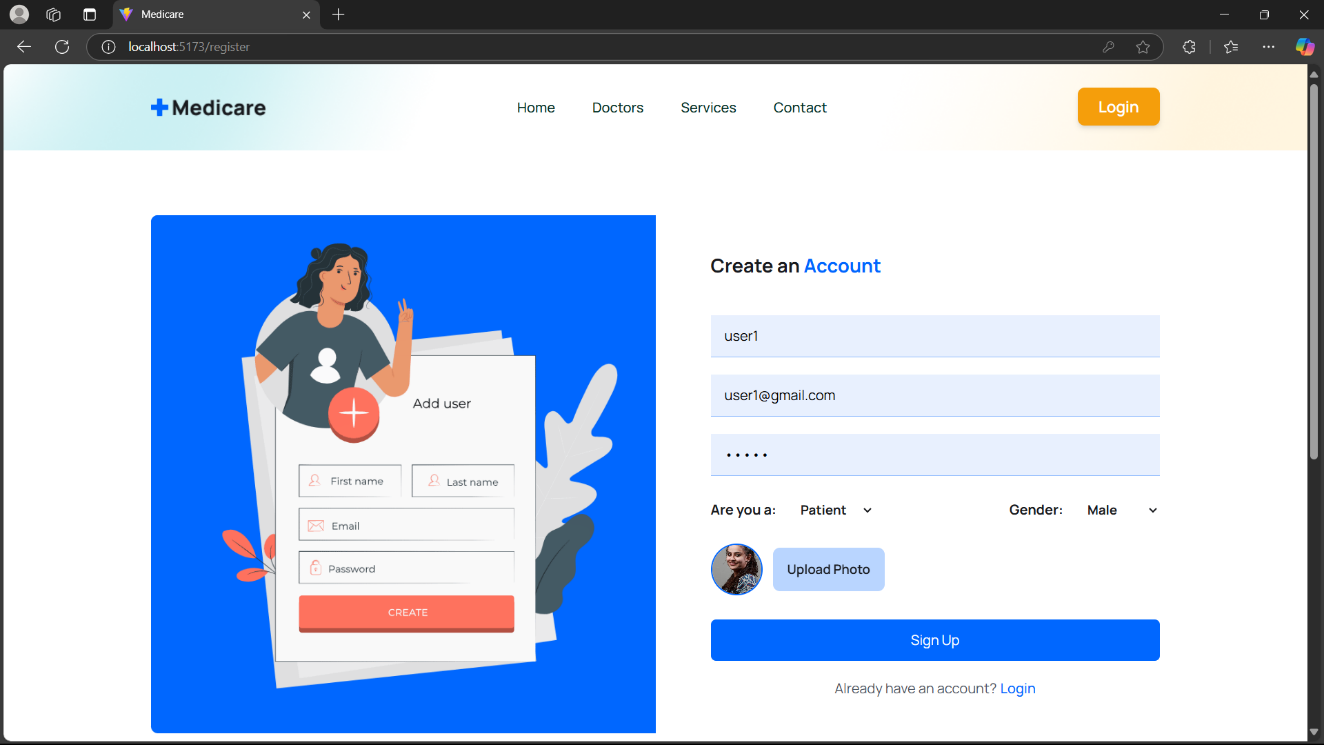
**Bug Tracking:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Bug ID | Bug Description | Steps to reproduce | Severity | Status | Additional feedback |
| BG-001 | Confirmation email not received | 1. Register as user 2. Wait for confirmation email | Medium | In Progress | Difficult in receiving emails due to SMTP misconfiguration and mail service throttling during peak requests. |
| BG-002 | Doctor dashboard not loading slots | 1. Login as doctor 2. Go to availability section 3. Click on date | High | In Progress | Possibly a database read delay |
| BG-003 | Payment API | 1. Open app on mobile 2. Click on booking  3. Click on payment | High | closed | The problem has been resolved by ensuring proper handling of asynchronous responses. |

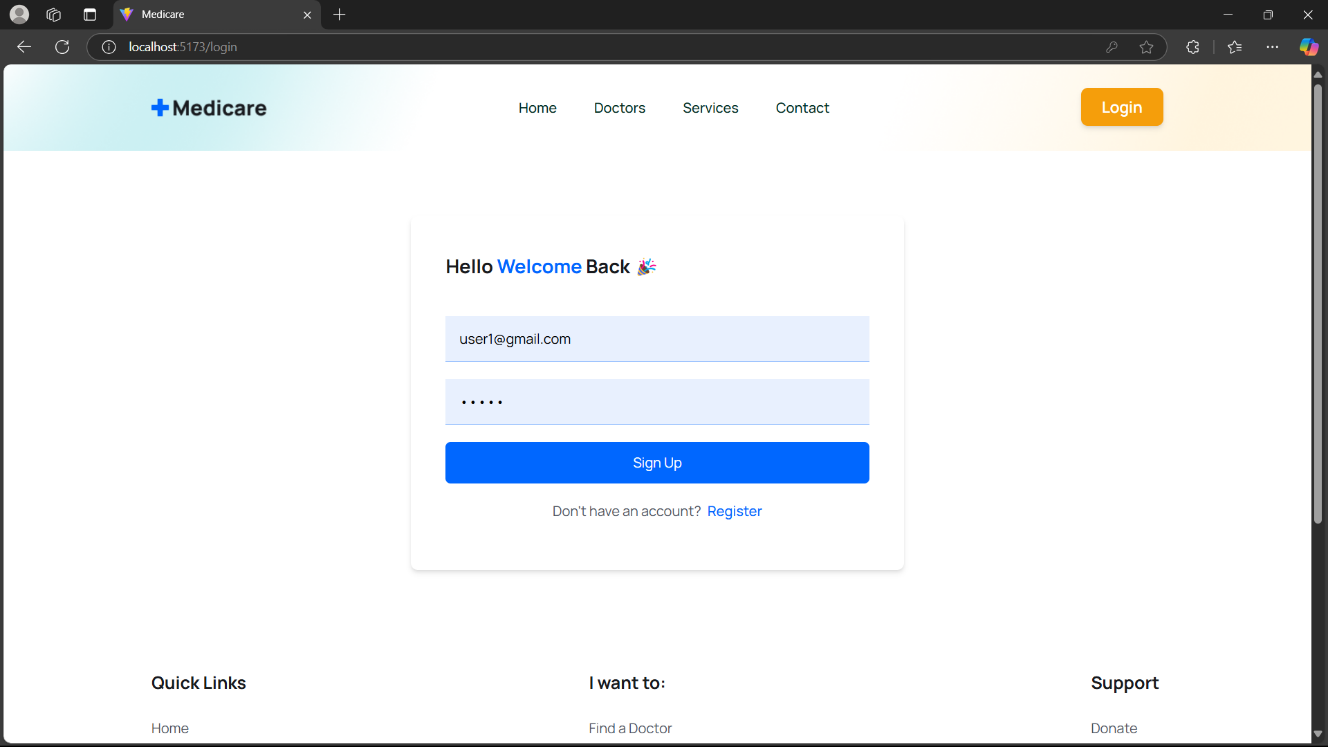
**7. RESULTS**

**7.1 Output Screenshots**

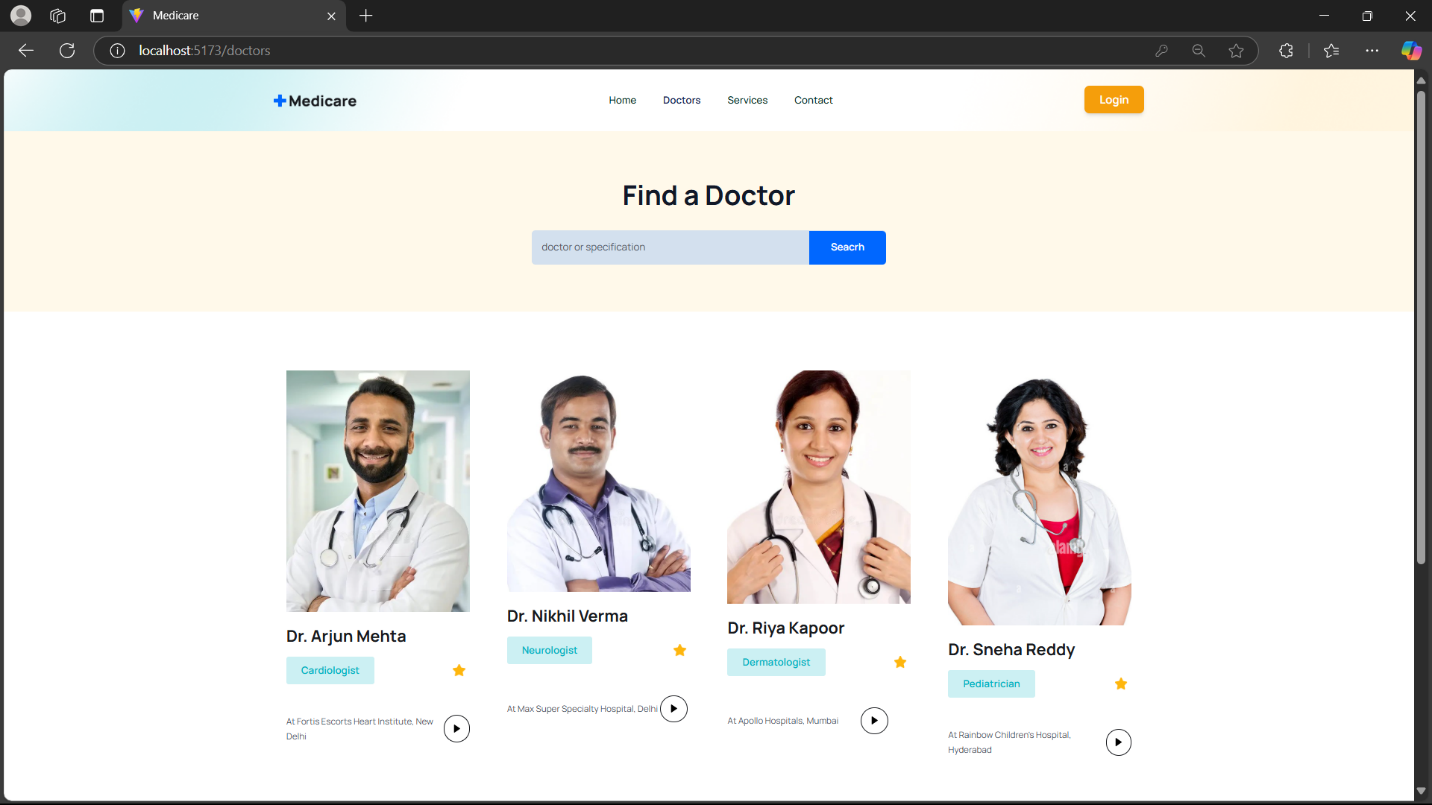
**Registration**



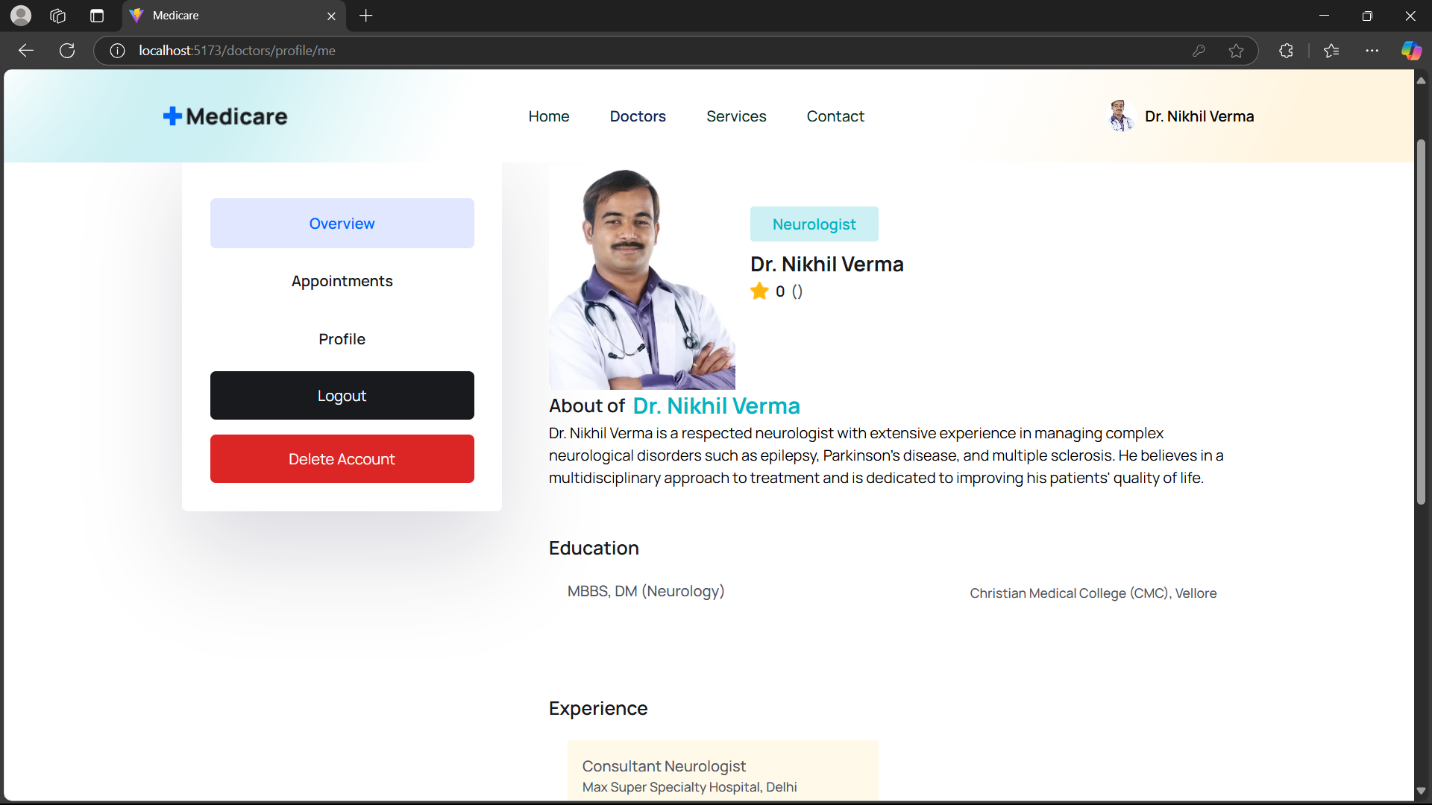
**Login**



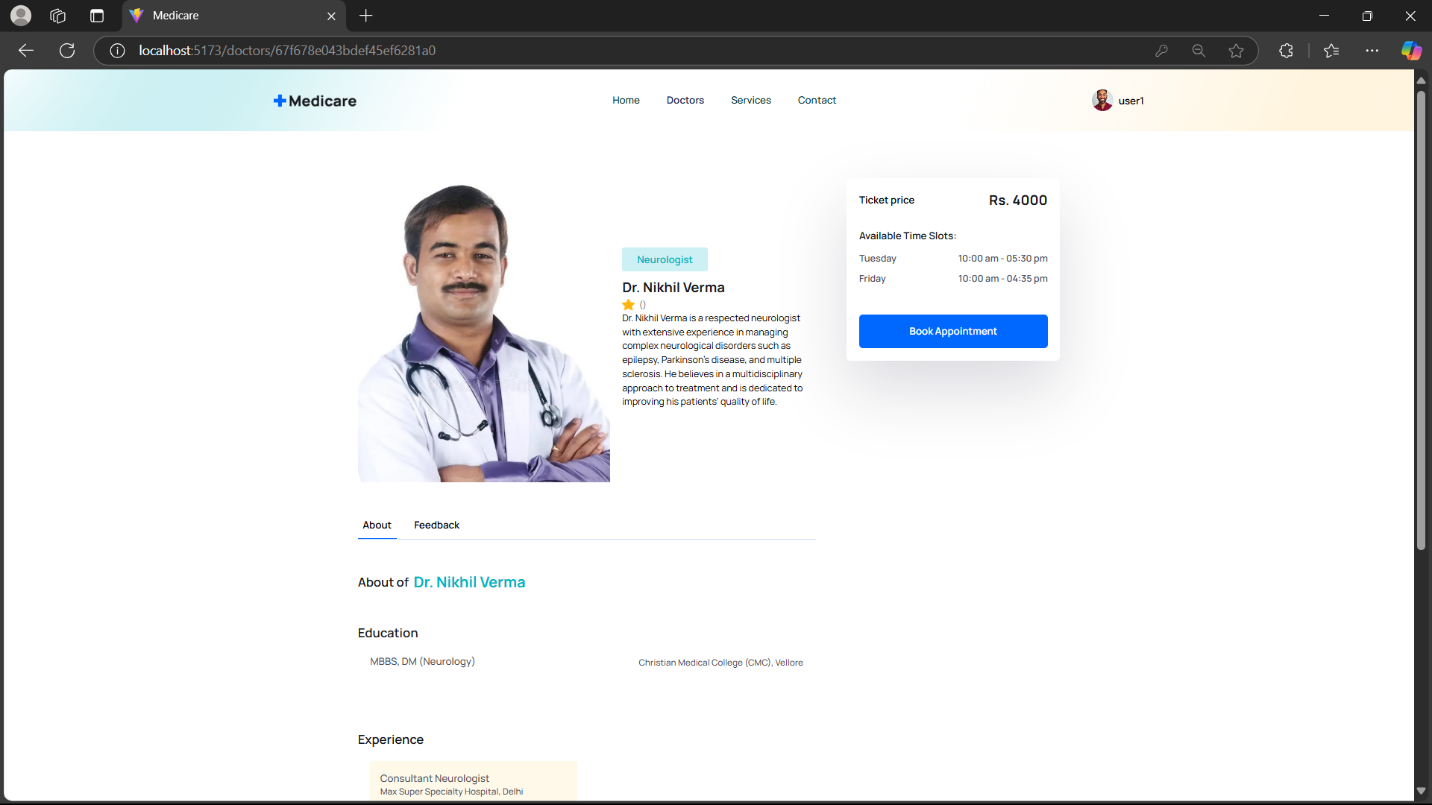
**Doctors**



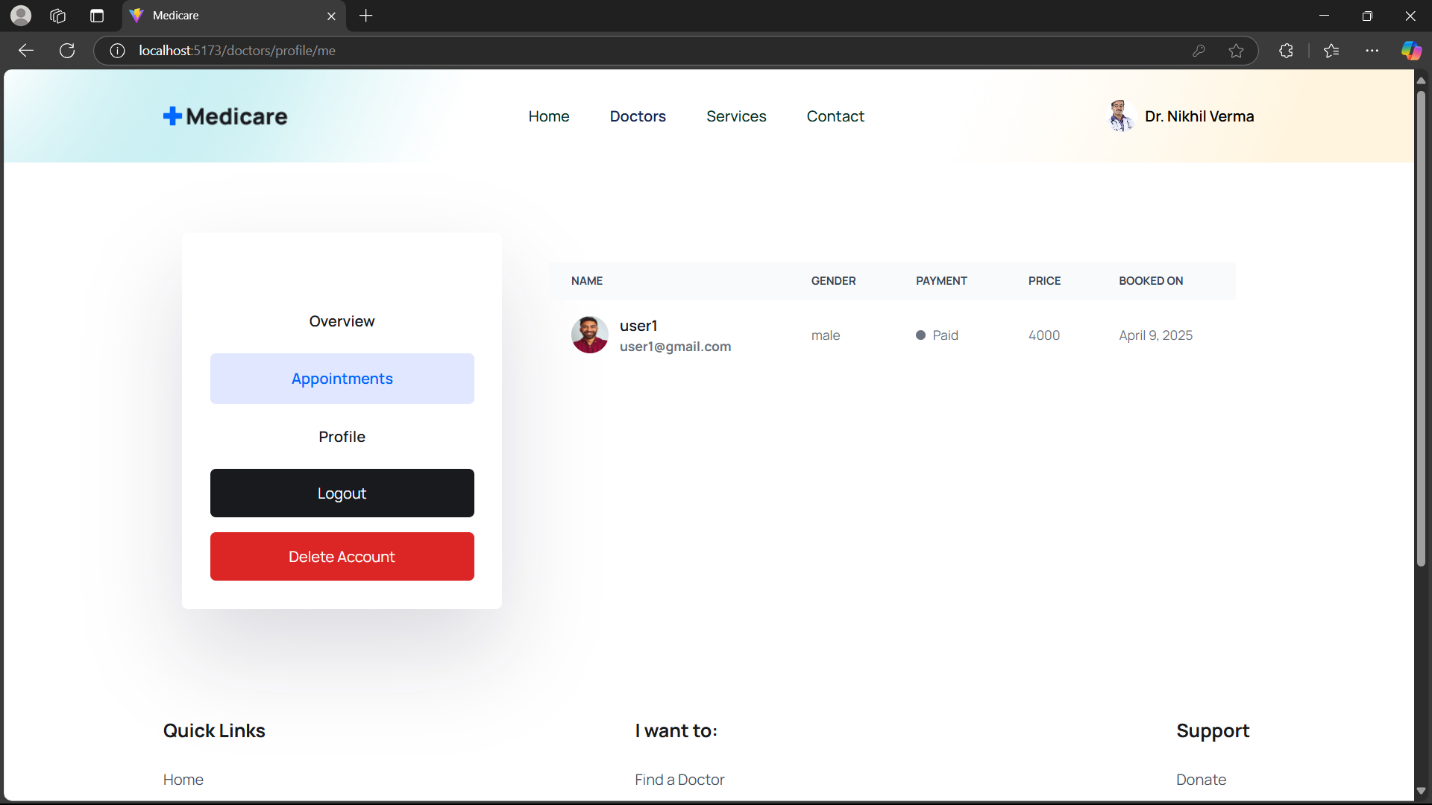
**Doctors Overview**



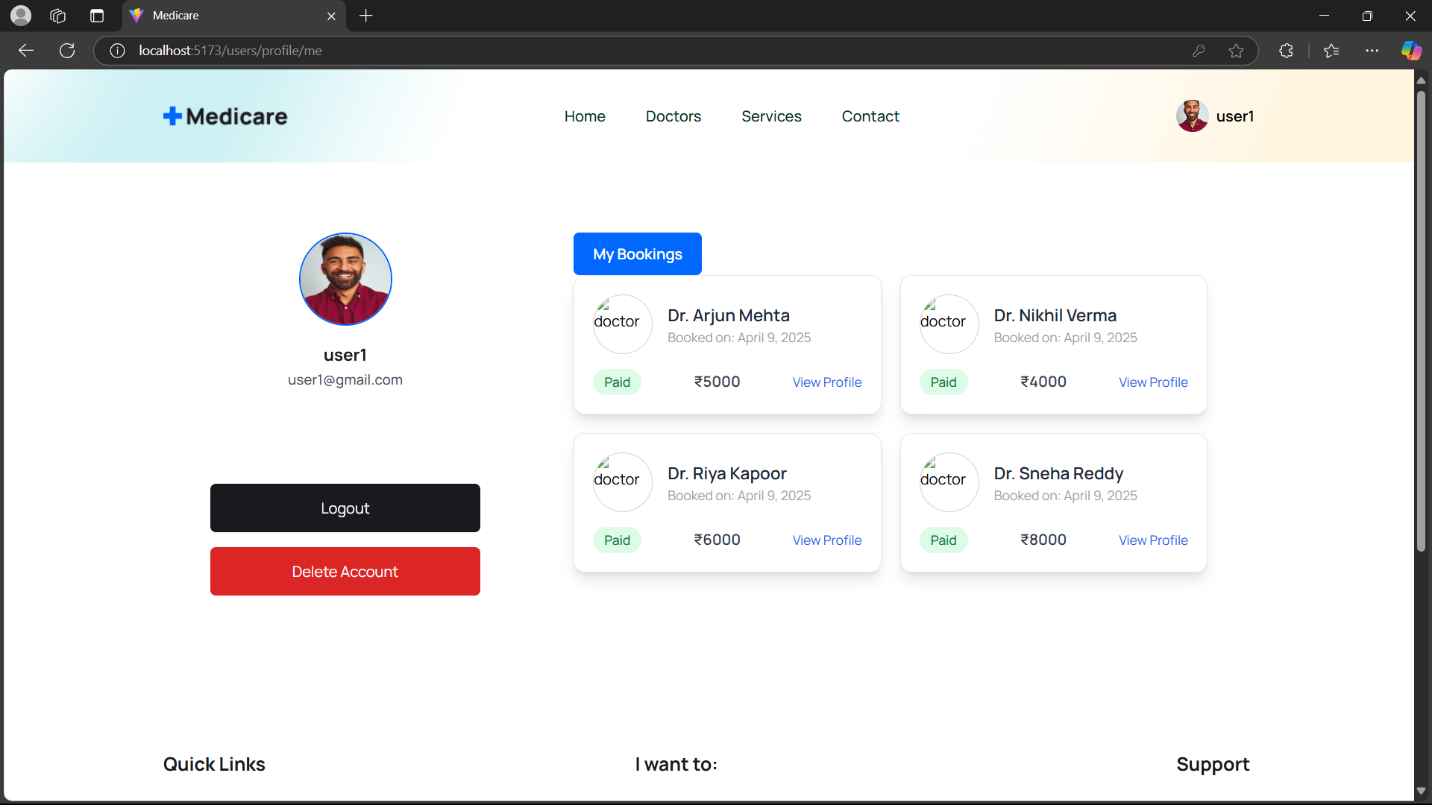
**About the Doctor**



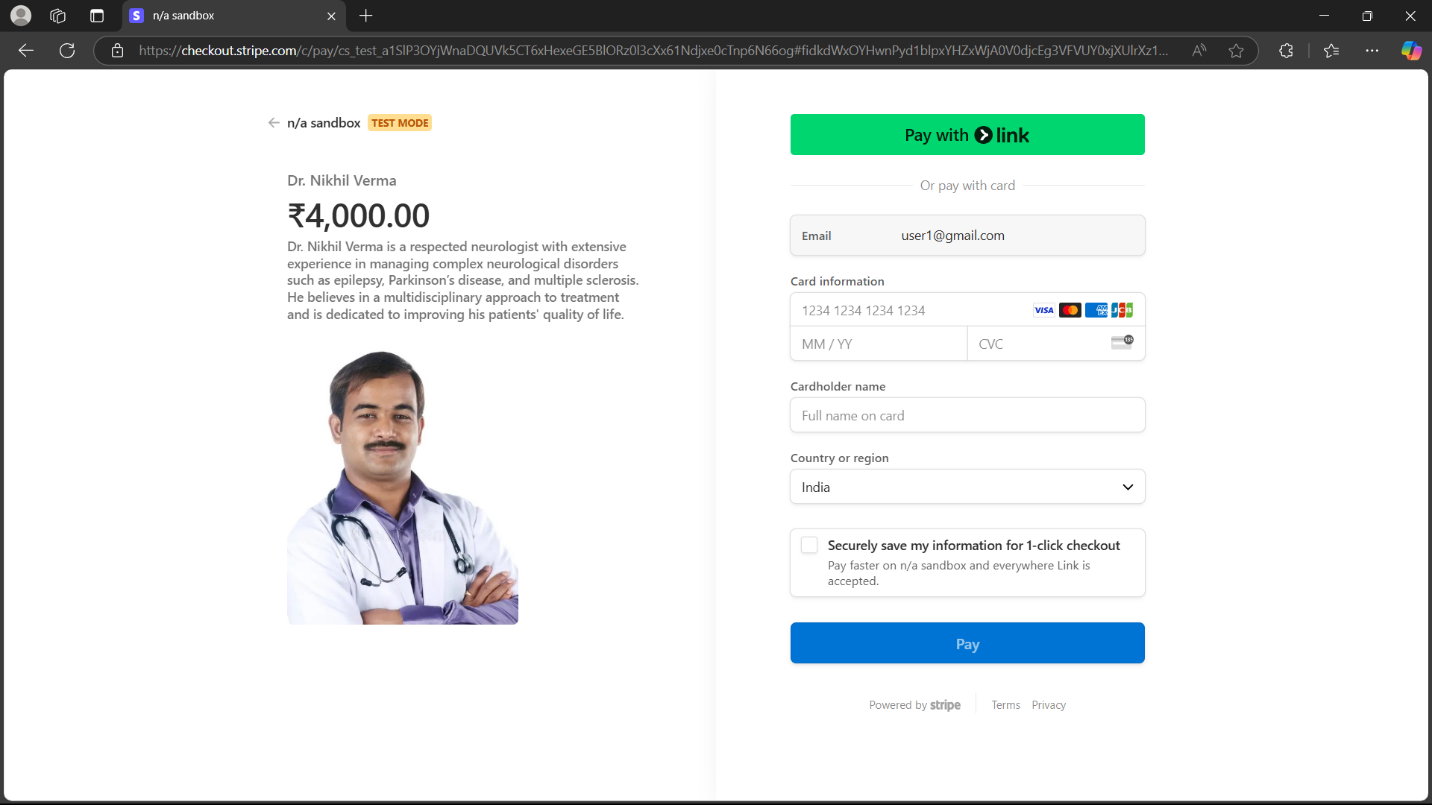
**Appointment**



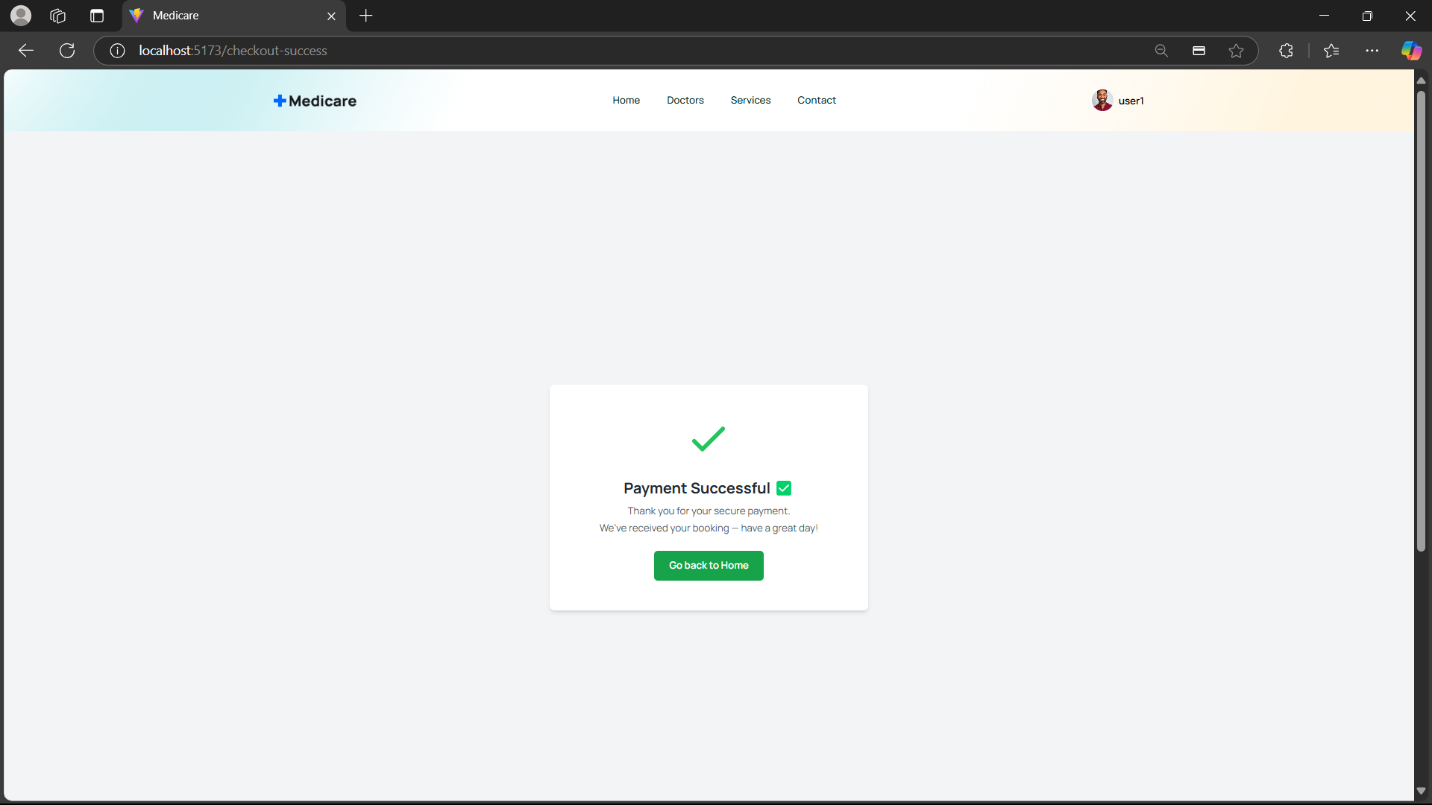
**User Bookings**



**Payment**



**Payment Successful**



**8. ADVANTAGES & DISADVANTAGES**

**Advantages**

* Patients can conveniently book appointments with doctors from anywhere at any time, reducing the need for physical visits.
* Each user—patient, doctor, or admin—has access to features specific to their role, improving security and ease of use.
* Doctors can view, manage, and update their appointment schedules efficiently from their dashboards.
* The system allows easy management of user data and appointments, reducing paperwork and improving data accessibility.
* The MERN stack enables fast, responsive interfaces and a scalable, maintainable backend structure.
* JWT ensures secure login and protects data from unauthorized access.
* A clean, responsive UI built with React ensures a smooth experience on desktops, tablets, and mobile devices.

**Disadvantages**

* A stable internet connection is required to access the platform, which may not be available in all regions.
* The current system focuses on appointment booking only and does not include features like real-time notifications or reminders.
* The application handles personal data, so proper privacy practices and compliance need to be ensured.
* Currently, the platform is only available via web browser; there is no dedicated mobile app version.

**9. CONCLUSION**

This project presents a web-based appointment booking system designed using the MERN stack to streamline the interaction between patients, doctors, and administrators. By offering a centralized platform for booking and managing appointments, the system eliminates the inefficiencies of manual scheduling and long waiting times, especially in areas with limited access to healthcare facilities.

Each user role—patient, doctor, and admin—is provided with a dedicated dashboard that supports specific functionalities tailored to their needs. Patients can book appointments with ease, doctors can manage their schedules more efficiently, and admins can oversee user management and platform activity. The use of modern technologies ensures the system is responsive, scalable, and secure.

Overall, this project serves as a practical solution for digitizing appointment management in healthcare settings, and it lays the foundation for further enhancements such as mobile app integration, notifications, or real-time updates in future versions.

**10. FUTURE SCOPE**

While the current system effectively handles appointment booking and user management, there are several areas for enhancement and expansion in future iterations of the project:

* Developing a mobile app version of the platform for Android and iOS would provide greater accessibility and convenience for users on the go.
* Integration of SMS or email notifications to remind patients and doctors of upcoming appointments or updates can improve user engagement and reduce missed bookings.
* Allowing doctors and patients to sync appointments with Google Calendar or Outlook can streamline scheduling and time management.
* Adding advanced search filters for specialties, availability, and locations can make it easier for patients to find the right doctor quickly.
* Incorporating multiple language options will make the platform more inclusive, especially for users in rural or diverse regions.
* An analytical panel with appointment trends, user activity, and system usage statistics can help admins make informed decisions.
* Implementing smart algorithms to suggest the best available slots based on user history and preferences.
* Allowing patients to rate and review doctors’ post-appointment can help maintain service quality and trust.

**11. APPENDIX**

**Source Code(if any)**

**Dataset Link**

**GitHub & Project Demo Link**

* **Demo Link :-**

<https://drive.google.com/file/d/1J5gXmthSkCDK6ui0DsVA4JbRiePr3sR1/view?usp=drive_link>

* **GitHub Link :-**

<https://github.com/Abhishek-Gangwar-18/SmartBridge_MERN_Project/tree/main>